Handling user-defined private contexts for location privacy in LBS

Maria Luisa Damiani
University of Milan, Italy
damiani@di.unimi.it

Marco Galbiati
University of Milan, Italy
mar.galbiati@gmail.com

ABSTRACT

We present a privacy-preserving framework for the protection of location from potentially untrustworthy location providers (LP), offering geolocation services to LBS subscribers, across indoor and outdoor settings. This framework, called *Placeprint*, is built on the metaphor of *private place*[1]. A private place is a user-defined spatial context which belongs to the personal sphere of an individual, e.g. home. In Placeprint, users equipped with commodity devices, can be geolocated in private places without revealing to the LP their presence. Moreover users can specify context-based privacy rules to forestall the disclosure of private places also to LBS providers. The ultimate goal is to provide users with the capability of exercising flexible control over the disclosure of the position to both LP and LBS provider.

Categories and Subject Descriptors

H.2.8 [**Database management**]: Database applications—*Spatial databases and GIS*; K.4.1 [**Computers and society**]: Public Policy Issues—*Privacy*

General Terms

Design, algorithms

Keywords

Privacy, location-based services, wi-fi, context-based privacy

1. INTRODUCTION

LBS provide spatially contextualized information services to location-aware mobile devices (clients). Very often the position is provided by a third party LP, e.g. Skyhook Wireless and Google Location Service. The client requests the position from the LP and then forwards such position along with the service request to the LBS provider (Figure 1). Typically, LPs offer geolocation services both indoor and outdoor, provided that the requester of the service transmits appropriate contextual information, e.g. Wi-Fi access points in proximity (e.g., $AP_1, ..., AP_n$). In a metropolitan region, with high density of Wi-Fi networks, the position can have an accuracy of 10-20 meters[1]. Note that users can interact with diverse LBS providers offering different services, therefore LPs are in a position to compile extensive records of users' location and movement. As LPs are potentially untrustworthy, privacy is seriously at stake. Unfortunately, existing privacy-preserving techniques, such as [2] do not protect the position from the LP but exclusively from the LBS provider; moreover, LPs do not generally accept anonymous communications (e.g., onion routing).

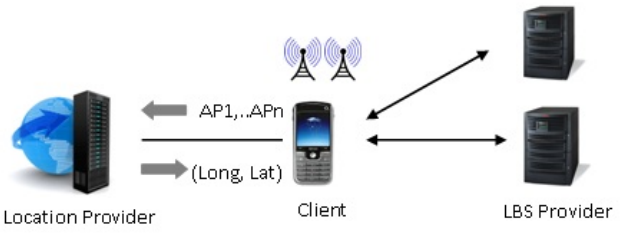


Figure 1: LBS Architecture: the client transmits to the LP contextual information (e.g. set of Wi-Fi access points, AP_1,..AP_n) to obtain the position

To more clearly motivate this privacy issue, consider the following scenario in which Alice is subscriber of a LBS. Alice is not concerned with the disclosure of her exact position except when she is at home and in few other places, such as the hospital where she is undertaking a medical treatment. Therefore when she is in one of those locations, Alice requires a coarse region to be disclosed in place of the true position. Imagine that Alice encodes this requirement in privacy rules, using for example a IETF GeoPriv-like privacy preference language [3]. A rule can be written:

$$Home \rightarrow CityOf(Home)$$

This rule means that whenever Alice is at *home* then the position to be disclosed is the city in which Alice lives, say the coordinates of Milan. The enforcement of this rule is seemingly straightforward: the client first requests its position from the LP; next, if such position is in proximity of home (assuming that home is assigned a position), the client

[1]http://www.skyhookwireless.com/howitworks/

ACM SIGSPATIAL GIS '12 Nov. 6-9, 2012. Redondo Beach, CA, USA

computes the coordinates of the city using for example a reverse geocoding service. Unfortunately, this approach leads to a privacy leak because the position is necessarily disclosed to the LP which computes it. Consider also that in this scenario, indoor positioning services can be only provided by the LP as users are assumed to use commodity devices and services.

Placeprint provides users with the capability of handling spatial contexts, such as home, in a privacy-preserving way. Contexts, once defined, are automatically recognized without revealing to the LP the user's presence in those places. Moreover users can define context-based privacy rules to specify the position to be transmitted to the LBS provider. These rules enable the integration of existing privacy preserving techniques, such as location obfuscation methods.

2. HANDLING PRIVATE PLACES

We argue that an approach to the protection of position from LP is to minimize the interaction with it. The motivating observation is that the amount of information that the user transmits to the LP exceeds what is really necessary to determine the users' position. For example every time a service is requested from a given place, e.g. home, the client transmits the same or similar contextual information, e.g. Wi-Fi access points. Based on this observation, we envision a solution in which clients acquire the capability of recognizing places that have been already visited. This way the position is only requested to the LP when it is strictly necessary. We confine the protection to a subset of positions, in particular those which can be associated with *private places*. Private place is an abstraction which conceptualizes the intuition that there are some regions of space that belong to the personal sphere. Private places are user-defined, e.g. home, myfriend, and are not confined to any specific setting, i.e. indoor/outdoor, or physical boundary. Whenever the user is in a private space, the position should not be disclosed to the LP. The technical challenge posed by the private place metaphor is recognizing whether the position is inside or outside a private place without interacting everytime with the LP. We refer to this operation as *place recogniton*. For example, techniques for dynamic place recognition such as [4, 5] enable the recognition of places that have been already visited based on the characteristics of the Wif-Fi and GSM infrastucture nearby. We take inspiration from these approaches to develop a solution for the recognition of private places.

Minimizing the interaction with the LP, however, does not forestall the disclosure of the private place to the LBS provider, e.g. every time Alice is at home, a position conventionally associated with the private place at the time the place is defined, is disclosed to the LBS provider (conversely the service could not be requested). Therefore if the LBS provider is untrustworthy or collude with the LP, location privacy is again at risk. To achieve a comprehensive protection of location from both the data collectors (LP and LBS provider), Placeprint enables the specification of context-based location transformation rules, globally defining the privacy policy of the user.

Section 2 illustrates the system architecture, Section 3 presents the interaction flow, i.e how users specify private places and privacy preferences, and what is achieved. The conclusive section reports future plans.

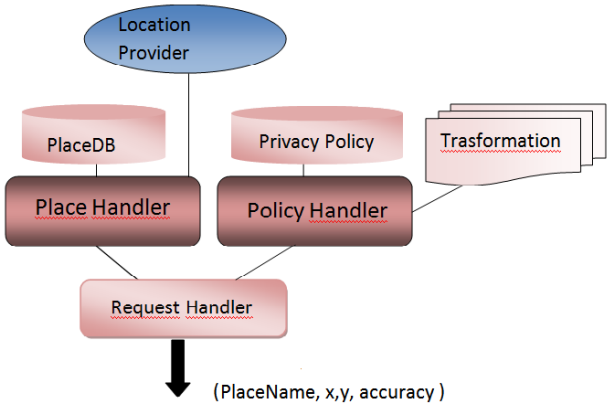


Figure 2: Placeprint architecture

3. ARCHITECTURE

Placeprint runs on the client. Figure 2 illustrates the system architecture. The Request Handler is the entry point of the system accepting and responding to geolocation requests. Geolocation requests are processed by two components, the Private Place Handler and the Privacy Policy Handler, respectively:

- The Private Place Handler is the key component enabling the construction and recognition of private places. Places are created upon explicit user's request by associating a name to a place signature defined in terms of GSM cells and Wi-Fi access points (hereinafter, *beacons*). Private places are then recognized by matching current context with one of the place signatures recorded in the Place DB. If the location is innocuous (i.e. not private), the system interacts with the LP to get the accurate coordinates while the returned placename is null.

- The Privacy Policy Handler enforces the context-based privacy rules stored in the Privacy Policy DB. It matches the outcome of the Place Handler against the privacy rules stored in the Privacy Policy DB and possibly applies position transformation functions, taken from a library of operations, to determine the location to be finally disclosed.

3.1 Private Place Handler

Key issue is defining the place signature. In [4, 5], a place signature is a response-rate histogram constructed by repeatedly scanning the networks in a time window. A place is then recognized if its signature is similar to one of the signatures previously built and then recorded. Unfortunately, constructing and thus also recognizing the place signature takes significant time (i.e. up to 10 seconds)). This is primarily due to the fact that the system needs to identify the whole set of beacons in proximity of a place, in a situation in which the radio signal is subject to noise and attenuation and thus beacons are listened intermittently. In our domain, place recognition has to be instantaneous, in order to not compromise the quality of the geolocation service. This motivates the investigation of a different approach, which sacrifices a little of accuracy for efficiency.

3.1.1 Creating a place signature

The idea is to define a place signature as set of *observations*, where an observation is the set of beacons that are detected at a certain position and instant. Formally an *observation* is a binary vector $\mathbf{o}=(b_1, b_2, ..b_n)$ in the n-dimensional beacon space where $b_i = 1$ if the i-esim beacon is detected and $b_i = 0$ otherwise. A signature s is a set of distinct observations, i.e. $s=\{o_1, .., o_q\}$ with $o_i \neq o_j$. In a metropolitan area where tens of beacons can be listened, observations do not repeat frequently, even at high sampling rates. The minimum number of observations in a signature is set experimentally (e.g. 20). Two place signatures are said to overlap when the same observation is present in both signatures. In order to ensure that places are distinguishable, we introduce the constraint that the maximum degree of pairwise similarity between the observations in the two signatures must be not greater than a threshold value $\tau \in [0, 1)$ (distinguishability constraint). For example if $\tau = 0$, the observations in the two signatures must not share any beacon. We use as similarity measure the Jaccard index applied to binary vectors (i.e. Tanimoto index), namely given two observations $o_i = (x_1, ..., x_m)$ and $o_j = (y_1, ..., y_m)$, the index $J \in [0, 1]$ is computed as follows:

$$J((x_1, ..., x_m), (y_1, ..., y_m)) = \frac{\sum_i (x_i \wedge y_i)}{\sum_i (x_i \vee y_i)}$$

Denoted with with d_j the Jaccard distance, i.e. 1-J, the distinguishability constraint between signatures s_1 and s_2 can be expressed as:

$$\min_{o_i \in s_1, o_j \in s_2} d_j(o_i, o_j) > \tau$$

The Jaccard distance is a metric. With little abuse of terminology we refer to the quantity $d(s_1, s_2) = \min_{o_i \in s_1, o_j \in s_2} d_j(o_i, o_j)$ as the *distance* between two places of signature s_i and s_j. We can thus say that two places are distinguishable if their distance is greater than τ. Signatures are stored in the PlaceDB. Each signature is associated with a representative point (x,y).

3.1.2 Recognizing private places

This operation takes in input the current observation o_c and returns a place name if the point is estimated to fall in such a place, otherwise null. A point is within a given place when the minimum distance between the *singleton signature* $s_c = \{o_c\}$, and the signature s of the place does not exceed threshold value $\delta \leq \tau$, i.e.

$$d(s_c, s) \leq \delta$$

The operation is performed as follows: it determines the set of signatures $H = \{s_i\}_i$ satisfying the above inequality, i.e. $d(s_c, s_i) \leq \delta$. If |H|=1, the place is found; if |H|=0, the position is innocuous; if |H|>1, the current observation is classified using k-nearest neighbor over the beacon vector space with metric the Jaccard distance. Denoted with $KNN = \{o_1, .., o_k\}$ the set of k closest observations to o_c (from any signature), the current observation is classified of place p if it holds:

$$p = argmax_{place} \sum_{o_i \in KNN} I(placeOf(o_i) = place)$$

where $placeOf(o_i)$ is the place the observation o_i refers to and I (e)=1 if e=true, 0 othw. Experimentally, the value of the parameter δ is set to $\delta = \tau = 0.3$.

3.1.3 Enhancing accuracy

The gain in efficiency is paid in terms of accuracy where accuracy, in our context, refers to the capability of recognizing when the user is inside or outside a private place. False positives are the positions classified as private even though they are innocuous. This result into loss of *spatial accuracy*. False negatives are the positions which are considered innocuous while in reality are private. This results into privacy loss. Three methods are proposed to enhance accuracy: a) Signatures are automatically upgraded with new observations once these observations are classified. As a result the more places are visited, the more accurate their recognition. b) Lowering the value of δ. This reduces the risk of false negatives at the expenses of spatial accuracy, therefore it requires appropriate tuning. c) Signatures are deliberately upgraded by users for example based on feedbacks provided by the system.

3.2 Privacy policy handler

If the user is detected in a private place, the Policy Handler can choose to perturb the location to transmit or even to suppress it, depending on the user's preferences encoded in the privacy policy. The rules of the privacy policies are expressed in a simple language which takes inspiration from the privacy preferences languages proposed for location sharing applications [6, 3]. A privacy rule takes the form:

$$r : context \rightarrow transformation$$

where: *context* is a place condition possibly coupled with a temporal condition; and *transformation* is a function mapping the coordinates provided by the Private Place Handler onto the final location or Null (i.e. location is suppressed). The set of rules $\{r_1, r_2, ..r_k\}$ forms the user's (privacy) policy.

A place condition can be: a) a user-defined private place; b) a system-defined place e.g. *PrivatePlace* and *Everywhere*. For example *home* is true if the user is estimated to be in the private place labeled home; *PrivatePlace* is true if the position falls in some private place, it does not matter which; *Everywhere* is true in any location (i.e. private and non private). A temporal condition is specified by the interval: $[< time >< time >]$ where *time* is a temporal value, e.g. hours and/or days of the week. A temporal condition is true if the current time is contained in the specified interval. In case of conflict among rules, the rule of the most specific place prevails. Further multiple rules can be defined for the same place over not-overlapping intervals. The language includes system-defined variables, e.g.(current) Point, (current) Place.

Transformations potentially includes a variety of operations. Simple transformations return a fake position or a coarse regions at predefined granularity. Non trivial obfuscation techniques might include, for example, semantic location cloaking [2]. Below some examples of privacy rules:

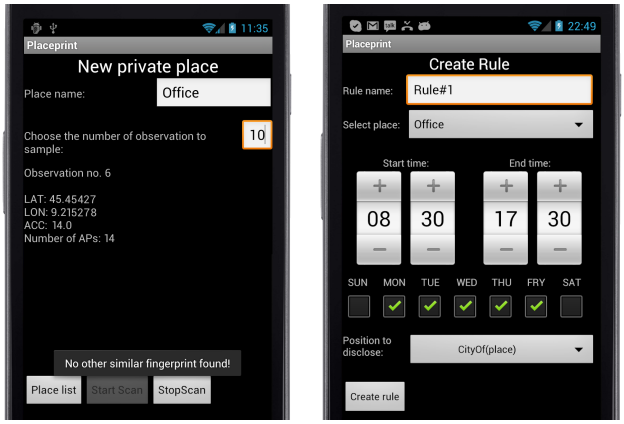
- *Home, [19:00, 08:00]* $\rightarrow$ *cityOf(Place)* specifies that when the user is at home in the specified time interval, the position is mapped onto the city of the private place.

- *PrivatePlace [19:00, 08:00]* $\rightarrow Null$ specifies that when the user is in a private place in the time interval, the

position is not disclosed (or is not meaningful). Note that this can be the default rule

- *Everywhere, [08:00, 19:00]* → *SemanticCloaking()* specifies that any position in the time interval is mapped onto the map generated by semantic location cloaking [2]. This map contains pre-computed cloaked regions enclosing sensitive places, e.g. hospitals.

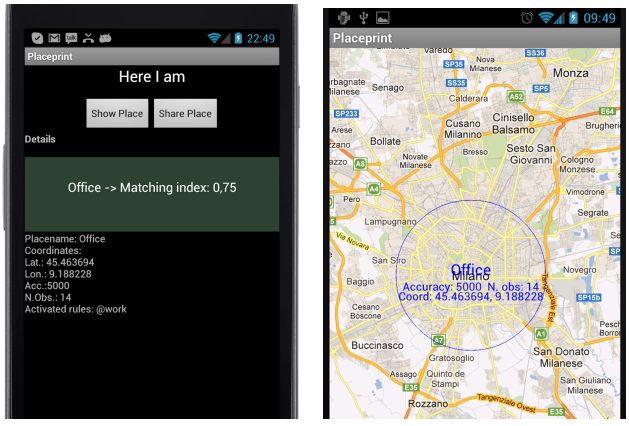
4. DEMO STORY LINE



(a) Creating a new private place (b) Creating a new rule

Figure 3: Privacy preferences

This demo illustrates the key functionalities of the Placeprint prototype. Developed in Java, the system runs on a device equipped with Android 2.3 (or higher). In particular, the demo shows how users can interact with the system to create places and privacy preferences, and share locations at varying and user-dependent granularity. The outcome of Placeprint is exemplified by the application *Here I am*.



(a) Requesting the location (b) Shared location

Figure 4: An application using Placeprint

- **Creating a private place.** From the user's perspective the operation is very simple (Figure 3). The user enters the name of the place, in this case *office*, and possibly the number of observations to acquire, then activates the acquisition of the place signature. The operation is fast. As a result, the representative position of the newly created place and summary data are displayed, including the number of beacons detected in proximity, e.g. 14 Wi-Fi Access Points.

- **Creating a privacy rule.** Figure 3.(b) shows the interface for entering the rule: the user specifies the rule name, e.g. *@work*, selects the place the rule is to be associated with from a list, e.g. office, possibly specifies a temporal constraint, e.g. working time, and finally selects the transformation from a list of operations, e.g. the function *cityOf(Place)*.

- **Here I am.** This application, illustrated in Figure 4.(a), displays the user's location returned by Placeprint. For example, if the true location is sufficiently close to the office, the place *Office* is recognized and the privacy rule @Work is activated. The matching index provides the user with information on place accuracy (i.e. the Jaccard Index). The outcome of the transformation operation is the location displayed in Figure 4.(b) at granularity of city. For demo purposes the location is named with the private place name. Actually, the shared location only consists of the pair *(point, accuracy)*.

5. CONCLUSIONS

Placeprint offers a first solution to the novel problem of protecting the user's location from untrusted LPs. The system integrates three different technologies, i.e. place learning, privacy policies and more conventional location privacy enhancing technologies, e.g. location obfuscation, to provide customizable protection against different parties. Future work will focus on conceptual extensions of both the place recognition method, e.g. to detect entrance and exit from private places, and the policy specification language.

6. REFERENCES

[1] M.L. Damiani. Third party geolocation services in LBS: privacy requirements and research issues. *Transactions on Data Privacy*, 4(2):55–72, 2011.

[2] M.L. Damiani, C. Silvestri, and E. Bertino. Fine-grained cloaking of sensitive positions in location-sharing applications. *IEEE Pervasive Computing*, 10(4):64–72, October 2011.

[3] Schulzrinne H., Tschofenig H., Morris J., Cuellar J., Polk J., and J. Rosenberg. Common Policy: A Document Format for Expressing Privacy Preferences. http://tools.ietf.org/html/rfc4745, 2007.

[4] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, and J. Hughes. Learning and recognizing the places we go. In *Proc. of the 7th international conference on Ubiquitous Computing*, UbiComp'05, 2005.

[5] Donnie H. Kim, Younghun Kim, Deborah Estrin, and Mani B. Srivastava. Sensloc: sensing everyday places and paths using less energy. In *Proc. of the 8th ACM Conf. on Embedded Networked Sensor Systems*, 2010.

[6] M. Benisch, P. G. Kelley, N. Sadeh, and L. Cranor. Capturing location-privacy preferences: quantifying accuracy and user-burden tradeoffs. *Personal Ubiquitous Comput.*, 15(7):679–694, 2011.